

Parallel implementation of electronic structure eigensolver using a partitioned folded spectrum method

E.L. Briggs

*Center for High Performance Simulation and Department of Physics,
North Carolina State University, Raleigh, North Carolina, 27695-7518*

C.T. Kelley

*Center for High Performance Simulation and Department of Mathematics,
North Carolina State University, Raleigh, North Carolina, 27695-8205*

J. Bernholc

*Center for High Performance Simulation and Department of Physics,
North Carolina State University, Raleigh, North Carolina, 27695-8205*

(Dated: March 2, 2015)

A parallel implementation of an eigensolver designed for electronic structure calculations is presented. The method is applicable to computational tasks that solve a sequence of eigenvalue problems where the solution for a particular iteration is similar but not identical to the solution from the previous iteration. Such problems occur frequently when performing electronic structure calculations in which the eigenvectors are solutions to the Kohn-Sham equations. The eigenvectors are represented in some type of basis but the problem sizes are normally too large for direct diagonalization in that basis. Instead a subspace diagonalization procedure is employed in which matrix elements of the Hamiltonian operator are generated and the eigenvalues and eigenvectors of the resulting reduced matrix are obtained using a standard eigensolver from a package such as LAPACK or SCALAPACK. While this method works well and is widely used, the standard eigensolvers scale poorly on massively parallel computer systems for the matrix sizes typical of electronic structure calculations. We present a new method that utilizes a partitioned folded spectrum scheme (PFSS) that takes into account the iterative nature of the problem and performs well on massively parallel systems. Test results for a range of problems are presented that demonstrate an equivalent level of accuracy when compared to the standard eigensolvers, while also executing up to an order of magnitude faster. Unlike $O(N)$ methods, the technique works equally well for metals and systems with unoccupied orbitals as for insulators and semiconductors. Timing and accuracy results are presented for a range of systems, including a 512 atom diamond cell, a cluster of 13 C60 molecules, bulk copper, a 216 atom silicon cell with a vacancy, using 40 unoccupied states/atom, and a 4000 atom aluminum supercell.

I. INTRODUCTION

Electronic structure calculations based on density functional theory are widely used in physics and materials science. These calculations usually involve numerical solutions of the Kohn-Sham equations for a set of eigenvalues and eigenvectors.

$$\mathbf{H}_{ks}[\psi_i] = -\frac{1}{2}\nabla^2\psi_i + V_{eff}\psi_i = \lambda_i\psi_i \quad i = 1, N \quad (1)$$

where

$$V_{eff} = V_{hartree} + V_{xc} + V_{ionic} \quad (2)$$

Self consistent pseudopotentials¹⁻³ of various forms are used to represent V_{ionic} while $V_{hartree}$ and V_{xc} are the exchange correlation and hartree potentials. A typical solution process proceeds by generating an initial set of electronic orbitals and an associated charge density. A linear combination of atomic orbitals from the constituent ions is often used for this together with a superposition of atomic charge densities as a starting density. The initial

wavefunctions may be represented by expansions in basis functions, such as plane waves^{4,5}, augmented waves⁶ or grid based discretizations⁷⁻⁹. The initial density is then used to compute exchange correlation and hartree potentials using functionals of the electron density¹⁰, which along with the ionic potentials form V_{eff} . This representation of V_{eff} is then substituted back into the Kohn-Sham equations that are solved for a new set of electronic orbitals. This process is repeated until self consistency is reached, where self consistency is defined as the difference between $V_{eff}(in)$ and $V_{eff}(out)$ potentials being less than a specified tolerance.

A variety of methods have been used to solve Eq.(1) for each self consistent step. Many of these methods include a subspace diagonalization and solution of a real symmetric eigenvalue problem as part of the procedure. For large systems this will normally be the most expensive part of the calculations as the work required scales as the cube of the number of electronic orbitals. Since this is the main computational bottleneck for large DFT calculations, considerable work has been carried out to either eliminate the cubic scaling terms entirely or reduce their prefactor. Algorithms that fall into the first category of

ten rely on the restricted spatial range made possible by real space methods. These include both grid based^{11,12}, and localized basis function approaches.¹³ The present work, which is a partitioned folded spectrum¹⁴ method (PFSM) falls into the second category and works by distributing eigenpairs over independent processing nodes. Eigenpairs on a given node are solved for using submatrix diagonalizations and an iterative folded spectrum technique is used to couple the submatrix solutions to the full matrix. No distinction is made between occupied and unoccupied states, in contrast to the spectrum slicing method developed by Schofield et al.¹⁵ where only the occupied eigenspectrum is divided into discrete slices. These are distributed over the nodes and filtering functions are used to restrict the eigenpairs on a specific slice to a discrete range of the occupied orbitals. The current method also shares some similarities with the shift and invert parallel spectrum transformation (SIPs) used by Zhang et al.¹⁶ for tight binding calculations, which generate sparse matrix eigenproblems as opposed to the dense matrices addressed in the current work. Their approach takes advantage of the sparsity to reduce the scaling factor from $O(N^3)$ to $O(N^2)$ for sufficiently sparse matrices.

Since direct diagonalization of Eq.(1) is difficult, a subspace diagonalization procedure is employed in which matrix elements of the Hamiltonian operator are generated. The eigenvalue problem for the resulting reduced matrix is then given by.

$$\mathbf{A}\mathbf{x}_i = \lambda_i\mathbf{x}_i \quad i = 1, N \quad (3)$$

This can be solved using a variety of standard library routines. These include LAPACK and/or MAGMA¹⁷ routines, which utilize a single processing node, and ScaLapack, which is designed to use multiple nodes. When running on multiple nodes of a supercomputer or cluster, LAPACK is a poor choice since it can only utilize the processing power of a single node. On a system with GPU's, the Magma libraries are a viable alternative for N ranging up to a few thousand. While Magma can currently only apply the processing power of a single node, GPU's are well suited for this type of problem and are so fast that solutions may be achieved in an acceptable amount of time. For problems where N is greater than a few thousand, ScaLapack usually becomes the best choice. The exact size of N depends on the relative balance of computational power to communications speed on the computer system being used. Unfortunately, for any given value of N a point will be reached where the performance of ScaLapack scales poorly as the number of nodes is increased due to communications overhead. Current implementations of ScaLapack are also unable to effectively utilize GPU's, leaving large amounts of processing power unused.

The current work is designed to address the shortcomings of these methods. The main design goal is to minimize the wall clock time required for a solution of Eq.(3). Exceptional efficiency in terms of the total operations

count is not necessarily needed to achieve this. Instead, the focus is on effectively using as much of the processing power available, as opposed to the current methods that may leave much of that power untapped on large clusters.

II. METHODOLOGY

Let $\{\lambda_i\}$ and $\{\mathbf{u}_i\}$ be the eigenvalues and eigenvectors of \mathbf{A} . We order λ_i in distance from the target ϵ . So

$$|\lambda_1 - \epsilon| < |\lambda_2 - \epsilon| \leq |\lambda_3 - \epsilon| \leq \dots |\lambda_n - \epsilon|. \quad (4)$$

The eigenvector corresponding to the eigenvalue closest to ϵ may be determined with the iteration.

$$\mathbf{x}^{n+1} = (\mathbf{I} - \alpha(\mathbf{A} - \epsilon\mathbf{I})^2)\mathbf{x}^n \quad (5)$$

This is just the power method for the matrix.

$$\mathbf{I} - \alpha(\mathbf{A} - \epsilon\mathbf{I})^2 \quad (6)$$

and it will converge to the desired eigenvalue

$$1 - \alpha(\lambda_1 - \epsilon)^2, \quad (7)$$

if the eigenvalues of $\mathbf{I} - \alpha(\mathbf{A} - \epsilon\mathbf{I})^2$ are ordered so that for each n

$$\alpha|\lambda_n - \epsilon| < 1 \quad (8)$$

and

$$\alpha < \frac{1}{|\lambda_n - \epsilon|}. \quad (9)$$

Then

$$|1 - \alpha(\lambda_n - \epsilon)^2| \leq |1 - \alpha(\lambda_{n-1} - \epsilon)^2| \leq \dots \leq |1 - \alpha(\lambda_2 - \epsilon)^2| < |1 - \alpha(\lambda_1 - \epsilon)^2| \quad (10)$$

for $1 \leq i \leq N - 1$.

If (9) is satisfied then given an initial estimate of an eigenvector \mathbf{x}_i^j and a shift value ϵ , the iterative process will produce an eigenvector with an eigenvalue close to ϵ .

$$\mathbf{x}_i^{j+1} = \mathbf{x}_i^j - \alpha(\mathbf{A} - \epsilon\mathbf{I})^2\mathbf{x}_i^j \quad 0 < \alpha < 1 \quad (11)$$

This may be implemented via a sequence of matrix-vector products which eliminates the need to explicitly square the matrix $\mathbf{A} - \epsilon\mathbf{I}$. Since the iterative process is only guaranteed to produce eigenvectors for eigenvalues close to the shift value ϵ , we must also have a way of

generating a suitable set of shifts and initial eigenvectors. We do this by solving a sequence of eigenvector problems generated from overlapping diagonal submatrices of the original matrix \mathbf{A} .

In particular, each diagonal submatrix \mathbf{B} is of order M where $M = \gamma N$ and $0 < \gamma \leq 1$. Since the work required to solve each submatrix scales as M^3 , each individual submatrix requires γ^3 less work. The total computational work required then depends on the magnitude of γ and the number of submatrices needed. Even when this is larger than the work required by a standard solver such as LAPACK or ScaLapack, the wall clock time required for a solution can be significantly less because the algorithm is better adapted for massively parallel computer architectures. By assigning individual submatrices to smaller blocks of processing nodes, the scalability constraints found in the standard solvers are alleviated. In our tests we have found for values of N ranging up to a few thousand (common in electronic structure calculations), assigning one submatrix per processing node works well on machines with GPU accelerators. For this case using a computing platform with P processing nodes, a slice of eigenvectors of size $m = N/P$ may be assigned to each node as shown in Fig.(1).

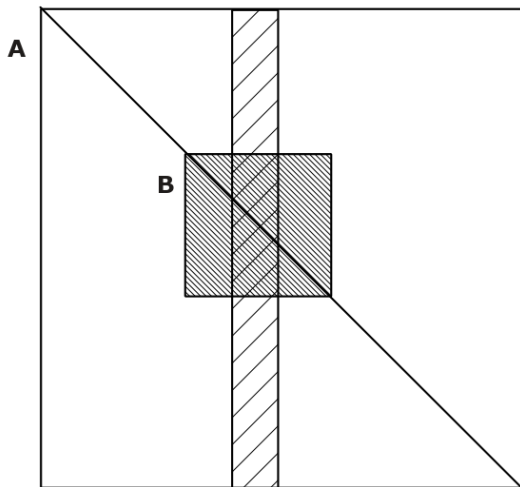


FIG. 1. The region with coarse grained cross hatching represents the set of eigenvectors assigned to an individual processing node. The region with fine grained cross hatching represents both submatrix \mathbf{B} and the set of eigenvectors associated with a solution of $\mathbf{B}\mathbf{y}_i = \lambda\mathbf{y}_i$

The eigenvector solutions \mathbf{y}_i for the submatrices are used to construct the starting vectors \mathbf{x}_i^0 for Eq.(11) and the corresponding eigenvalues are used as the shifts ϵ . Since $M < N$, the elements of \mathbf{x}_i that do not have a corresponding component in the submatrix solution are set to zero. The iterative process from Eq.(11) is then applied and after all eigenvectors have been processed, Gram-Schmidt orthogonalization is used to ensure or-

thonormality of the resulting eigenvectors. Note that since this process is applied on a subspace and is readily parallelized, it represents only a moderate fraction of the total work.

This method will not work reliably for an arbitrary matrix. This may be demonstrated by taking a set of eigenvectors that satisfy Eq.(1) and randomly mixing them to generate \mathbf{A} . In this case the eigenvectors of the submatrices will consist of linear combinations of eigenvectors from the full spectrum and the iterative process described in Eq.(11) can produce nearly identical estimates for the submatrix λ_i that will not span the full spectrum.

A key condition for success then is that \mathbf{A} be diagonally dominant. We have generally found that using standard diagonalization for the first 3 to 5 SCF steps is sufficient when the submatrix widths are 30 percent of the full width of \mathbf{A} and the number of eigenvectors computed via Eq. (11) from each submatrix is less than 10 percent of the full width. We thus expect major savings during the usual applications of DFT, such as geometry optimization and ab initio molecular dynamics, in which only the first ionic step would require some standard diagonalizations, while PFSM would be used for the many subsequent steps.

III. ACCURACY AND CONVERGENCE

A variety of tests were performed to determine both accuracy and the effect on convergence rates of the new method. The RMG^{7,9} code developed at NCSU was used for all tests. The first system consists of a 512-atom diamond cell with a Vanderbilt¹⁸ ultrasoft pseudopotential used to represent the ionic cores. Test runs using submatrix widths ranging from 0.1 to 0.3 show that a width of 0.3 produces results identical to those obtained using standard full diagonalization. While the size of this system is typical of production SCF calculations, it is a perfect crystal with highly degenerate eigenvalues. Many systems of interest are disordered and we expect that they should actually be easier to handle computationally because the degeneracies will normally be broken in such systems.

As an example of a disordered system we selected a cluster of 13 C60 molecules in a non-equilibrium FCC structure. This system again exhibits identical results between standard full diagonalization and the new method once the submatrix width reaches 0.3.

Since diamond and c60 both have bandgaps, bulk copper is selected as an example of a metallic system. Orbital occupations near the Fermi level are assigned using a Fermi-Dirac distribution with $\mu = 0.04$ eV. In this case the overall convergence rate was slower, as expected for a metal, but the submatrix width required in order for the PFSM results to match those from full diagonalization is reduced from 0.3 to 0.2. A similar reduction of the required submatrix width has also been observed with other metallic systems, which we attribute to the break-

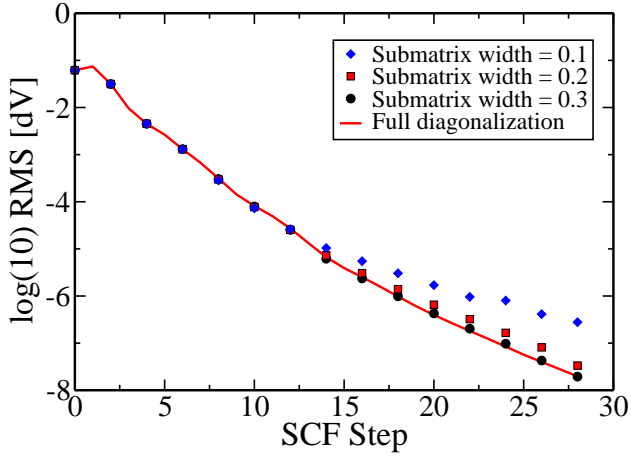


FIG. 2. Convergence rates for a 512 atom diamond cell for

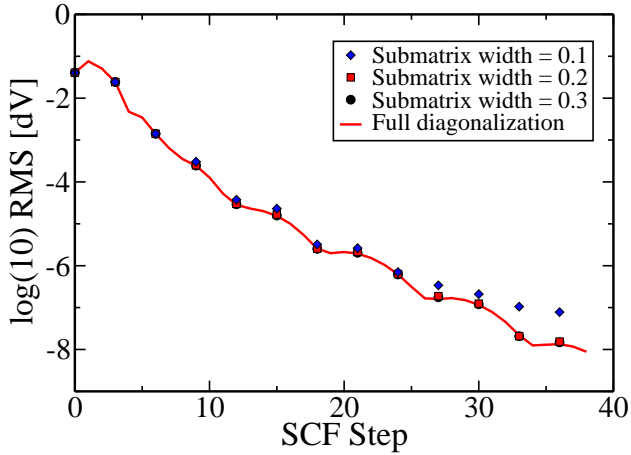


FIG. 3. Convergence rates for a cluster of 13 C60 molecules (780 atoms total) for various submatrix widths.

ing of degeneracies due to the Fermi-Dirac distribution of the orbital occupations.

IV. PERFORMANCE AND PARALLELIZATION

The PFSM method can use different types of eigensolvers for the submatrices and RMG currently supports LAPACK and MAGMA. Improvements in execution time depend on problem size and type, and the architecture of the target platform. As the bulk of the execution time for the RMG code is normally spent in the multigrid preconditioner, which exhibits $O(N^2)$ scaling, and the eigensolver that exhibits $O(N^3)$ scaling, we expect that the overall speedup will increase as N increases and this is indeed observed. The results presented in the following tests use the Bluewaters system, a Cray XK7 located at NCSA, where each processing node consists of a multi-core Opteron CPU and a single Nvidia K20X GPU. The nodes are connected via a 3-D torus, with each link pro-

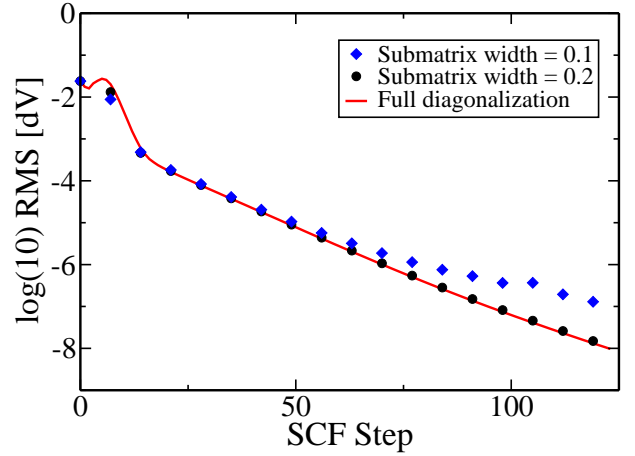


FIG. 4. Convergence rates for a 256 atom copper cell for various submatrix widths.

viding up to 9.6 GBytes/sec of bidirectional bandwidth. The current version of PFSM used in the RMG code is targeted towards systems where $N \leq 10000$. This limit is not specific to the algorithm, but rather is a consequence of the current implementation that assigns each submatrix to a single processing node. As this problem size is quite common in electronic structure calculations, the method is widely applicable.

Several test systems were selected to illustrate the potential performance gains. The timings depend on the submatrix width, which is always selected so that convergence rates and final results are essentially identical for PFSM and full diagonalization. Results for the 256 atom copper cell that were used in the convergence tests illustrated in Fig.(4) are presented in Table I. The improvement in the eigensolver is a factor of 2.28 for MAGMA and 14.16 for LAPACK. The overall speedup factors are 1.08 and 2.72 respectively.

	MAGMA	ScaLapack	LAPACK	MAGMA PFSM	LAPACK PFSM
Total time	5.93	6.52	15.36	5.46	5.64
Eigensolver	1.23	1.83	10.34	0.54	0.73

TABLE I. Time per SCF step (seconds) for 256 atom copper cell with 256 unoccupied orbitals, $N=1664$, using 64 Cray XK7 nodes.

Results for the next system are presented in Table II. These results were generated for a 216 atom silicon vacancy with approximately 40 unoccupied states per atom. This provides a good example of the performance improvements possible for systems with a large number of unoccupied states, which are common in GW calculations. In contrast to the previous result, where the MAGMA eigensolver was faster than ScaLapack, we now see better performance from ScaLapack. This is not surprising since the MAGMA result is being generated by a single GPU which is saturated for this value of N . The Eigensolver portion of the MAGMA-PFSM result is in

turn more than 5 times faster than the pure MAGMA result and 2.6 times faster than the ScaLapack result. We also note that the eigensolver results for LAPACK-PFSM are more than 20 times faster than the LAPACK result and indeed are nearly twice as fast as ScaLapack.

	MAGMA	ScaLapack	LAPACK	MAGMA PFSM	LAPACK PFSM
Total time	69.07	43.39	317.23	25.30	30.40
Eigensolver	52.04	26.75	296.67	10.36	14.23

TABLE II. Time per SCF step (seconds) for 216 atom silicon vacancy 40 unoccupied orbitals/atom, N=9136, using 64 Cray XK7 nodes.

The last test case presented in Table III is a 4000 atom aluminum cell running on 512 Cray XK7 nodes. The number of unoccupied states per atom is less than 1, which leads to a value of N that is only slightly larger than in the previous system, even though the cell volume is 14.9 times larger. The Eigensolver portion of the ScaLapack calculation is 2.35 times faster than the MAGMA result, while the MAGMA-PFSM result is 6.15 times faster than the pure MAGMA result.

	MAGMA	ScaLapack	LAPACK	MAGMA PFSM	LAPACK PFSM
Total time	64.75	40.28	351.16	25.27	35.78
Eigensolver	45.25	19.27	323.17	7.35	12.40

TABLE III. Time per SCF step (seconds) for 4000 atom aluminum supercell, N=9216, using 512 Cray XK7 nodes.

V. IMPLEMENTATION

The matrix elements of Eq(1) are generated as shown below.

$$\langle \psi_i | H_{ks} | \psi_j \rangle = \lambda_i \langle \psi_i | \psi_j \rangle \quad i, j = 1, N \quad (12)$$

This produces a generalized eigenvalue problem.

$$\mathbf{A}\mathbf{x}_i = \lambda_i \mathbf{B}\mathbf{x}_i \quad i = 1, N \quad (13)$$

Which is normally solved using a standard LAPACK or ScaLapack routine such as DSYGVD or PDSYGVD. These routines convert the generalized form into the standard form of Eq.(3) before solving for the eigenpairs. We use a similar approach in the PFSM but take advantage of specific characteristics of the problem to do the conversion efficiently on massively parallel computer architectures. Conversion to standard form is equivalent to multiplying both sides of Eq.(13) by \mathbf{B}^{-1} . The explicit computation of \mathbf{B}^{-1} is not required however and since \mathbf{B} is diagonally dominant the identity serves as a good

initial approximation, ($\mathbf{B} \neq \mathbf{I}$ at convergence due to ultrasoft pseudopotentials⁹). The initial guess may be improved using an iterative process. Let

$$\mathbf{Z}_{n+1} = (\mathbf{I} - \mathbf{D}^{-1}\mathbf{B})\mathbf{Z}_n + \mathbf{D}^{-1}\mathbf{A}\mathbf{x} \quad (14)$$

where \mathbf{D} is the diagonal of \mathbf{B} . Then at convergence

$$\mathbf{Z} = \mathbf{Z} - \mathbf{D}^{-1}\mathbf{B}\mathbf{Z} + \mathbf{D}^{-1}\mathbf{A}\mathbf{x} \quad (15)$$

therefore $\mathbf{B}\mathbf{Z} = \mathbf{A}\mathbf{x}$ and $\mathbf{Z} = \mathbf{B}^{-1}\mathbf{A}$. This iteration has the desirable property that the individual columns of \mathbf{Z} can be computed independently. Given P computational nodes, we assign N/P columns of \mathbf{Z} to each node and achieve excellent scalability. After the completion of this step an MPI_Allgather call is used to ensure that all processing nodes have a copy of \mathbf{Z} that is now used in Eq(3).

The matrix \mathbf{Z} obtained in Eq.(15) is then decomposed over processing nodes in the manner illustrated in Fig.(1). While the current implementation only uses MAGMA or LAPACK running on a single node to solve each submatrix, this is not an actual limitation of the method. Groups of processing nodes running ScaLapack or some other form of distributed eigensolver could also be used to solve the submatrices, work is under way to implement this. We expect that this will only be advantageous when the value of N exceeds 15,000, because for submatrix sizes of $M \leq 3000$ (corresponding to full matrix sizes of 10000 to 15000), MAGMA running on a single node is the fastest option for computing the submatrix solutions on the Cray XK7. One consideration when using MAGMA is that the submatrix size is constrained by the amount of GPU memory available on a single node. For the XK7 this is 6GB and a considerable fraction of that memory is reserved for other datastructures. Consequently, the present RMG implementation is currently restricted to values of $N \leq 10000$. We expect to expand this to values of $N \approx 35,000$ for the XK7 via improved memory management. Future architectures, such as the Summit system scheduled to be installed at ORNL in 2017, will have a larger coherent memory space accessible to the GPU accelerators (up to 512 GBytes/node), and should be able to handle much larger systems.

VI. SUMMARY

We have developed a new method, referred to as PFSM, for computing the eigenpairs of matrices of the type typically found in *ab-initio* electronic structure calculations. PFSM exploits specific characteristics of the problem, and the available hardware architectures to achieve up to an order of magnitude improvement in the eigensolver execution time. The improved speed can be obtained with no sacrifice in accuracy or convergence rates, and is particularly well suited to metallic systems.

The source code for PFSM is currently available in the latest stable release of RMG, which may be obtained at <http://rmgdft.sourceforge.net>. It is integrated into the main code base and work is underway to move it into a separate library with a calling interface similar to the standard eigensolvers. This should make it easy to incorporate the method into other electronic structure codes

with minimal work.

VII. ACKNOWLEDGEMENTS

We gratefully acknowledge discussions with Drs. Miroslav Hodak and Wenchang Lu. EB was supported by NSF grant ACI-1339844, CTK by NSF ACI-1339844 and DMS-1406349, and JB by DOE DE-FG02-98ER45685.

-
- ¹ Joel A. Appelbaum and D. R. Hamann, “Self-consistent pseudopotential for si,” *Phys. Rev. B* **8**, 1777–1780 (1973).
 - ² D. R. Hamann, M. Schlüter, and C. Chiang, “Norm-conserving pseudopotentials,” *Phys. Rev. Lett.* **43**, 1494–1497 (1979).
 - ³ Alex Zunger and Marvin L Cohen, “First-principles nonlocal-pseudopotential approach in the density-functional formalism: Development and application to atoms,” *Physical Review B* **18**, 5449 (1978).
 - ⁴ Marvin L. Cohen, M. Schlüter, James R. Chelikowsky, and Steven G. Louie, “Self-consistent pseudopotential method for localized configurations: Molecules,” *Phys. Rev. B* **12**, 5575–5579 (1975).
 - ⁵ R. Car and M. Parrinello, “Unified approach for molecular dynamics and density-functional theory,” *Phys. Rev. Lett.* **55**, 2471–2474 (1985).
 - ⁶ Peter E. Blöchl, “Generalized separable potentials for electronic-structure calculations,” *Phys. Rev. B* **41**, 5414–5416 (1990).
 - ⁷ E. L. Briggs, D. J. Sullivan, and J. Bernholc, “Real-space multigrid-based approach to large-scale electronic structure calculations,” *Phys. Rev. B* **54**, 14362–14375 (1996).
 - ⁸ James R Chelikowsky, N Troullier, and Y Saad, “Finite-difference-pseudopotential method: Electronic structure calculations without a basis,” *Physical review letters* **72**, 1240 (1994).
 - ⁹ Miroslav Hodak, Shuchun Wang, Wenchang Lu, and J Bernholc, “Implementation of ultrasoft pseudopotentials in large-scale grid-based electronic structure calculations,” *Physical Review B* **76**, 085108 (2007).
 - ¹⁰ John P Perdew, Kieron Burke, and Matthias Ernzerhof, “Generalized gradient approximation made simple,” *Physical review letters* **77**, 3865 (1996).
 - ¹¹ J. L. Fattebert and J. Bernholc, “Towards grid-based $o(n)$ density-functional theory methods: Optimized nonorthogonal orbitals and multigrid acceleration,” *Phys. Rev. B* **62**, 1713–1722 (2000).
 - ¹² J.-L. Fattebert and F. Gygi, “Linear-scaling first-principles molecular dynamics with plane-waves accuracy,” *Phys. Rev. B* **73**, 115124 (2006).
 - ¹³ José M Soler, Emilio Artacho, Julian D Gale, Alberto García, Javier Junquera, Pablo Ordejón, and Daniel Sánchez-Portal, “The siesta method for ab initio order-n materials simulation,” *Journal of Physics: Condensed Matter* **14**, 2745 (2002).
 - ¹⁴ Lin-Wang Wang and Alex Zunger, “Solving schrödingers equation around a desired energy: Application to silicon quantum dots,” *The Journal of chemical physics* **100**, 2394 (1994).
 - ¹⁵ Grady Schofield, James R Chelikowsky, and Yousef Saad, “A spectrum slicing method for the kohn–sham problem,” *Computer Physics Communications* **183**, 497–505 (2012).
 - ¹⁶ Hong Zhang, Barry Smith, Michael Sternberg, and Peter Zapol, “Sips: Shift-and-invert parallel spectral transformations,” *ACM Trans. Math. Softw.* **33** (2007), 10.1145/1236463.1236464.
 - ¹⁷ Emmanuel Agullo, Jim Demmel, Jack Dongarra, Bilel Hadri, Jakub Kurzak, Julien Langou, Hatem Ltaief, Piotr Luszczek, and Stanimire Tomov, “Numerical linear algebra on emerging architectures: The plasma and magma projects,” in *Journal of Physics: Conference Series*, Vol. 180 (IOP Publishing, 2009) p. 012037.
 - ¹⁸ David Vanderbilt, “Soft self-consistent pseudopotentials in a generalized eigenvalue formalism,” *Phys. Rev. B* **41**, 7892–7895 (1990).